

```
In [1]: import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
from sklearn.neighbors import KNeighborsClassifier
```

```
In [2]: df = pd.read_csv("7458_emails.csv")
```

```
In [3]: df.head()
```

Out[3]:

Email No.	the	to	ect	and	for	of	a	you	hou	...	connevey	jay	valued	lay	infra
0 Email 1	0	0	1	0	0	0	2	0	0	...	0	0	0	0	0
1 Email 2	8	13	24	6	6	2	102	1	27	...	0	0	0	0	0
2 Email 3	0	0	1	0	0	0	8	0	0	...	0	0	0	0	0
3 Email 4	0	5	22	0	5	1	51	2	10	...	0	0	0	0	0
4 Email 5	7	6	17	1	5	2	57	0	9	...	0	0	0	0	0

5 rows × 3002 columns



```
In [4]: df.isnull().sum()
```

Out[4]:

Email No.	0
the	0
to	0
ect	0
and	0
..	..
military	0
allowing	0
ff	0
dry	0
Prediction	0
Length:	3002, dtype: int64

```
In [5]: X = df.iloc[:,1:3001]
X
```

Out[5]:

	the	to	ect	and	for	of	a	you	hou	in	...	enhancements	connevey	jay	v
0	0	0	1	0	0	0	2	0	0	0	...	0	0	0	
1	8	13	24	6	6	2	102	1	27	18	...	0	0	0	
2	0	0	1	0	0	0	8	0	0	4	...	0	0	0	
3	0	5	22	0	5	1	51	2	10	1	...	0	0	0	
4	7	6	17	1	5	2	57	0	9	3	...	0	0	0	
...	
5167	2	2	2	3	0	0	32	0	0	5	...	0	0	0	
5168	35	27	11	2	6	5	151	4	3	23	...	0	0	0	
5169	0	0	1	1	0	0	11	0	0	1	...	0	0	0	
5170	2	7	1	0	2	1	28	2	0	8	...	0	0	0	
5171	22	24	5	1	6	5	148	8	2	23	...	0	0	0	

5172 rows × 3000 columns

In [6]: Y = df.iloc[:, -1].values
Y

Out[6]: array([0, 0, 0, ..., 1, 1, 0], shape=(5172,))

In [7]: train_x, test_x, train_y, test_y = train_test_split(X, Y, test_size = 0.25)

In [8]: svc = SVC(C=1.0, kernel='rbf', gamma='auto')
C here is the regularization parameter. Here, L2 penalty is used(default). It is
As C increases, model overfits.
Kernel here is the radial basis function kernel.
gamma (only used for rbf kernel) : As gamma increases, model overfits.
svc.fit(train_x, train_y)
y_pred2 = svc.predict(test_x)
print("Accuracy Score for SVC : ", accuracy_score(y_pred2, test_y))

Accuracy Score for SVC : 0.8979118329466357

In [14]: X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 0.2, random_s

In [15]: knn = KNeighborsClassifier(n_neighbors=7)

In [16]: knn.fit(X_train, y_train)

Out[16]: KNeighborsClassifier(n_neighbors=7)

```
In [17]: print(knn.predict(X_test))
```

```
[0 0 1 ... 0 1 0]
```

```
In [13]: print(knn.score(X_test, y_test))
```

```
0.8685990338164251
```

```
In [ ]:
```